VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra Informatiky

Univerzální dálkové ovládání PC

Universal IR Controller for PC

2010 / 2011 Petr Chromec

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Petr Chromec**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Univerzální dálkové ovládání PC
Universal IR Controller for PC

Zásady pro vypracování:

Předmětem práce je vytvořit jednoduchý IR přijímač připojitelný k PC, díky kterému bude možno toto PC ovládat za použití běžných dálkových ovládačů. Součástí musí být klientská aplikace, která bude schopna suplovat např. stisk klávesových zkratek atp. Aplikace může být navrhnuta jako víceuživatelská s různou definicí akcí k jednotlivým IR signálům.
V práci musí být obsaženo především:
1. Výběr vhodného obvodu pro příjem IR signálu.
2. Návrh a implementace uživatelské aplikaci umožňující ovládání PC.
3. Uživatelská a programátorská příručka.
4. Část věnující se relizaci a testování.

Seznam doporučené odborné literatury:

Podle pokynů vedoucího bakalářské práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Marian Mindek, Ph.D.**

Datum zadání: 19.11.2010
Datum odevzdání: 06.05.2011

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Místopřísežné prohlášení studenta

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě ....

podpis studenta

Poděkování

Na tomto místě bych rád poděkoval všem, kteří mi při tvorbě bakalářské práce pomáhali. Zejména pak vedoucímu práce panu Ing. Marianu Mindkovi, Ph.D. za cenné připomínky a odborné rady. Dále pak panu Ing. Martinovi Milatovi za pomoc při výrobě mého zařízení a rovněž za odborné rady a připomínky. A nakonec panu Ing. Igoru Českovi za informace, které zveřejnil na svých internetových stránkách.

Abstrakt

Bakalářská práce „Univerzální dálkové ovládání PC“ se zaměřuje na ovládání funkcí operačního systému Windows pomocí běžných dálkových ovladačů. Obsahuje čtyři hlavní části. První část se zabývá výběrem vhodného obvodu, jeho zhotovení, ceně a možnostem jaké toto zařízení nabízí. Druhá část pojednává o implementaci uživatelské aplikace, použitých algoritmech a popisu jednotlivých tříd programu. Třetí část je uživatelská příručka k aplikaci, umožňující pohodlné ovládání operačního systému z dálky pomocí dálkového ovladače. Je zde také popis jednotlivých funkcí a možností tohoto programu. Poslední část se věnuje testování uživatelské aplikace na různých platformách, operačních systémech a také na použití různých dálkových ovladačů v praxi.

Klíčová slova

Dálkový ovladač, příjem infra červeného kódu, ovládání funkcí operačního systému.

Abstract

Bachelor thesis „Universal IR Controller for PC“ is aimed at controlling the functions of operating system Windows using common remote controls. It contains four main parts. The first part covers selection of a suitable circuit, its manufacturing, cost, and what capabilities the device offers. The second part deals with the implementation of user application, the algorithms used and a description of each class of the program. The third part is a user guide to the application, allowing easy control of the operating system from a distance by remote control. There is also a description of the features and capabilities of this program. The last part is devoted to testing user application on different platforms, operating systems and also use of different remotes in practice.

Key Words

Remote control, receiving IR code, controlling the functions of operating system.

Seznam použitých zkratek a symbolů

AVR	- Alf (Egil Bogen), Vegard (Wollan) Risc procesor
cca	- circa
č.	- číslo
DLL	- Dynamic-link Library
DOS	- Disk Operating System
DPS	- deska plošných spojů
EEPROM	- Electrically Erasable Programmable Read-Only Memory
GUI	- Graphical User Interface
IR	- Infra Red
IS	- Isolated Storage
Kč	- koruna česká
KHz	- kilohertz
kg	- kilogram
LPT	- Line Print Terminal
LED	- Light-Emitting Diode
MB	- megabyte
např.	- například
OS	- operační systém
PC	- Personal Computer
RISC	- Reduced Instruction Set Computing
RS232	- Recommended Standard 232
SDK	- Software Development Kit
SMD	- Surface Mount Device
SP	- Service Pack
URL	- Unique Resource Locator
USB	- Universal Serial Bus
V	- volt
WMP	- Windows Media Player
XML	- Extensible Markup Language

Obsah

1. Úvod

Předmětem práce je vytvořit jednoduchý IR přijímač připojitelný k PC, díky kterému bude možno toto PC ovládat za použití běžných dálkových ovládačů. Součástí bude klientská aplikace, která bude schopna suplovat např. stisk klávesových zkratek atp. Aplikace může být navrhnuta jako víceuživatelská s různou definicí akcí k jednotlivým IR signálům.

Práce obsahuje:

1. Výběr vhodného obvodu pro příjem IR signálu

2. Návrh a implementace uživatelské aplikaci umožňující ovládání PC

3. Uživatelská a programátorská příručka

4. Část věnující se realizaci a testování

- Aplikace bude více méně nezávislá na typu dálkového ovládání

- Možnost provozovat více druhů ovladačů

- Aplikace by měla umožnit ovládání různých programů

2. Výběr vhodného obvodu pro příjem IR signálu

Rozhodl jsem se pro vytvoření obvodu a jeho propojení s počítačem přes rozhraní USB. V současnosti je USB rozhraní velmi populární zejména mezi uživateli. Je to způsobeno jeho jednoduchostí vzhledem ke koncovým uživatelům (Plug and Play, bez nutnosti restartu počítače). Je zde však potřeba programové podpory na straně počítače - ovladače zařízení.

Moje řešení je implementace USB do formy mikrokontroléru pomocí emulovaného USB protokolu přes firmware mikrokontroléru. Problémem při návrhu byla rychlost mikrokontroléru. Rychlost sběrnice USB je totiž vysoká: LowSpeed - 1.5MB / s, FullSpeed - 12MB / s, HighSpeed - 480Mbit / s. Existují sice i mikrokontroléry s vyššími rychlostmi, ale ty jsou již obtížněji dostupné, a to nejen cenově. Také jsou rozměrnější, mají hodně pinů, a to znamená složitější konstrukci. Proto jsem se rozhodl pro mikrokontrolér ATTINY2313, který by mohl stíhat LowSpeed USB. Na vyšší rychlosti USB zatím toto řešení nestačí, na zpracování jednoho "bitu" z USB linky je třeba totiž několik taktů mikroprocesoru: načtení, otestování, uložení operace. ATTINY2313-20PU je RISC mikrokontrolér z produkce Atmel - rodina AVR. Mají jednu instrukci na takt hodin. Navíc jejich instrukční sada a architektura je mnohem více bližší RISC. Kvůli synchronizaci s USB jsem použil přetaktování na 12MHz (AT90S23x3, použité panem Ing. Českem (Příloha 14) a také ATTINY2313, použité mnou, jsou původně pouze na 10MHz). Takto jsem získal více výkonu a 12MHz krystaly jsou lehce sehnatelné (např. vůči 10.5MHz, což je také násobek LowSpeed USB 1.5MB / s).

2.1 Konstrukce

Při mém řešení je hardware velmi jednoduchý a levný (cca 200,- Kč). Celou inteligenci zabezpečí obslužný firmware. Konstrukce je řešena jako infračervené dálkové ovládání počítače přes USB.

Toto zařízení umožňuje:

- Příjem infračerveného kódu (časový diagram přijatého kódu; posílá se i po sériové lince v reálném čase), příjem infračerveného kódu dělá mikroprocesor bez účasti počítače, to pro počítač znamená nulové vytížení.

- Ovládání 8 bitové vstupně-výstupní brány (každý bit může být nezávisle vstup nebo výstup, mohou se nezávisle řídit i pull - up rezistory na vstupních pinech).
- Čtení a zápis do interní 128 bytové EEPROM (uchování dat i při výpadku napájení - pro uživatele je přístupná celá paměť).
- Vyslání nebo přijetí znaku po sériové lince.
- Změnu rychlosti sériové linky v rozsahu cca 4800 Baud až 700 000 Baud (po zapnutí je to 57600 Baud).

Dostupné rychlosti jsou 4800, 9600, 19200, 38400 a 57600. Obslužná DLL automaticky detekuje neplatné rychlosti.

2.2 Verze první

Schéma hardware s mikroprocesorem AT90S2313-10 (ATTINY2313).

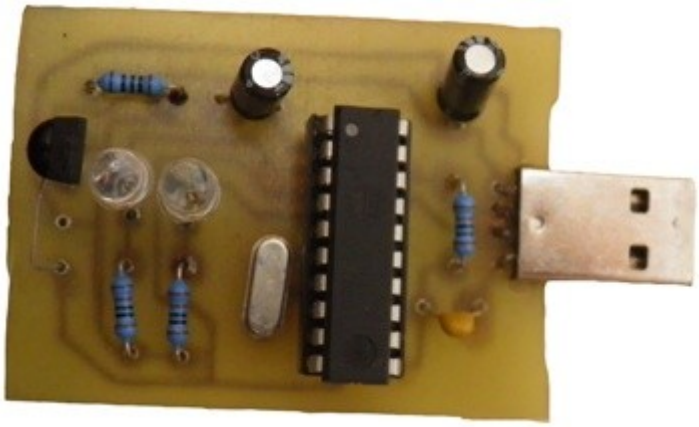
Toto původní schéma bylo vytvořeno v roce 2003 Ing. Igorem Českem a bylo mou hlavní inspirací. V řešení jsem použil ATTINY2313, protože AT90S2313-10 se dnes již nevyrábí. A k načtení IR signálu jsem použil přijímač SFH5110-36, který má vstupní napětí 5V a reaguje na frekvenci 36KHz. Pro frekvenci 36KHz jsem se rozhodl, protože jsem se dočetl, že frekvence 36KHz je mezi dálkovými ovladači nejrozšířenější.[1] Mé konkrétní schéma bylo upravené schéma (Obrázek 1) pro USB klíčenky a je podrobně vidět na následujících obrázcích (Obrázek 2, 3, 4).

Mé řešení a realizace vyšlo na druhý pokus (Obrázek 5), při prvním leptání byly některé cesty málo zakryté barvou a leptáním se přerušily. Druhý pokus byl úspěšný a po osazení součástkami a následném vložení hotového hardware (Obrázek 6) do PC jsem se pokusil nainstalovat ovladač, napsaný Ing. Igorem Českem, který je volně k dispozici na jeho stránkách[2]. Při vložení hardwaru do PC sice našel neznámé USB, ale při instalaci počítač s Windows Vista havaroval a počítač s Windows XP hlásil, že ovladač neobsahuje informace o daném hardwaru.

Později jsem zjistil, že zařízení bylo zkonstruováno špatně. A to proto, že jsem se do té doby nesetkal s SMD součástkami, pro které bylo schéma (Obrázek 4) nakresleno. SMD součástky se totiž letují na cuprextit na stranu mědi a ne na opačnou, jako je tomu u běžných

[1] URL: <http://www.ustr.net/infrared/infrared1.shtml> [cit. 2011-07-18]
[2] URL: <http://www.cesko.host.sk/download.php> [cit. 2011-07-18]

součástek. Proto jsem zařízení osadil mikrokontrolérem z opačné strany a z toho důvodu bylo zařízení nefunkční a nerozpoznáno žádným operačním systémem.



Obrázek 6 – hotová první verze zařízení

Celková cena první verze zařízení je 228, 40 Kč. (Tabulka 1)

2.3 Verze druhá

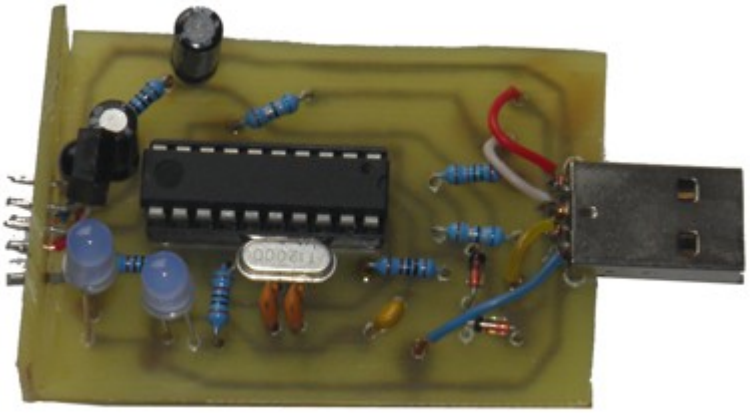
Proto jsem hledal a snažil se najít řešení tohoto problému a zjistil jsem, že v roce 2003, kdy vzniklo původní schéma, bylo USB o něco tolerantnější a pracovalo na 5V. Na nových PC je to však jiné. Výchozí napětí USB je sice stále 5V, ale komunikace probíhá pouze na 3.3V. Našel jsem tedy nové schéma (Obrázek 7, 8, 9) z roku 2009, které by mělo vyhovovat i novějším standardům USB. Mikroprocesor zůstal stejný (ATTINY2313-20PU), ale nahradil jsem IR přijímač SFH5110-36 za TSOP34838, který má vstupní napětí 2,7V – 5V a přijímá frekvenci 38KHz, která by měla při své odchylce zvládnout všechny používané frekvence dálkových ovladačů, ty jsou 36KHz – 40KHz (40KHz používají například ovladače SONY).

S výrobou zařízení podle tohoto schématu, jsem měl značné komplikace. Hlavně ve fázi leptání (Obrázek 10), kdy se mi vždy nějaká cesta vyleptala víc a cuprextit byl pak nepoužitelný.

Problém byl pravděpodobně v kvalitě použitého značkovacího fixu, který dostatečně nezakryl měď před leptacím roztokem (Chlorid železitý – $FeCl_3$).

Nakonec se mi však leptání přece jen povedlo a zařízení jsem osadil součástkami a připojil k počítači (Obrázek 11). Výsledek byl však stejný jako u prvního pokusu, a to ze stejného důvodu. Schéma totiž pořád vycházelo z SMD verze mikrokontroléru, z toho vyplývá,

že čip byl daný na špatné straně, měl tudíž obráceny piny, a proto byl nefunkční a zároveň jsem v této fázi přišel na další chybu, a to že mikrokontrolér nebyl nijak nastaven. Myslel jsem si, že stačí mikrokontroléru ponechat základní nastavení, to však není pravda. Bylo třeba nastavit správně jeho pojistky a nahrát obslužný program. To jsem se dočetl na internetových stránkách pana Ing. Česka, kde jsem zároveň nalezl i daný program pro mikrokontrolér a informace o tom, jak správně nastavit pojistky. Toto vedlo k dalšímu problému. Jak dostat program do mého mikrokontroléru?



Obrázek 11 – hotová druhá verze zařízení

Celková cena druhé verze zařízení je 237, 10 Kč. (Tabulka 2)

2.4 Výroba programátoru

Potřeboval jsem programátor, kterým bych mohl dostat do svého mikrokontroléru obslužný program a nastavit pojistky na správné hodnoty. A tak jsem našel na internetu schéma LPT programátoru (Obrázek 12), který podle popisu měl být schopný naprogramovat mimo jiné i mikrokontrolér ATTINY 2313.

Tento programátor je kabelem (Obrázek 14) připojen k počítači do LPT portu a má externí napětí 9V, které je usměrněno regulátorem napětí (U1) na 5V, což je vstupní napětí mikrokontroléru. Připojení mikrokontroléru k programátoru je možno realizovat dvěma způsoby, a to zaprvé přímo do patice programátoru vložit mikrokontrolér anebo do patice připojit pomocný kabel (Obrázek 15) a k samotnému mikrokontroléru připojit již pouze konektory na příslušné piny.

Výroba programátoru byla docela jednoduchá a téměř bez komplikací. (Obrázek 16, 17)

Jediný problém byl opět v otočení patice mikrokontroléru. Tu jsem dal znovu na špatnou stranu, a proto byl můj programátor zdánlivě nefunkční. Po vložení čipu, připojení baterie (externí zdroj 9V) a připojení programátoru k LPT portu počítače nebyl čip rozeznán ani programem PonyProg 2000 ani WinAVR, které se k programování mikokontrolérů používají. Zkoušel jsem je na různých operačních systémech, včetně DOS boxu. Ale díky otočené patici, tedy převrácení pinů mikrokontroléru se nic nestalo. Myslel jsem si tedy, že mám buď špatně sestaven svůj programátor, nebo špatně nastaven LPT port počítače. Zkoušel jsem různá nastavení BIOSu i různé počítače, ale nic nepomohlo. Proto jsem zkusil svůj problém vyřešit jinak.

V tom mi vyšli vstříc pan Ing. Martin Milata a pan Ing. Petr Olivka. Kterým tímto moc děkuji. Zapůjčili mi školní programátor a spolu s panem Ing. Milatou a nakonec jeho osobním programátorem jsme úspěšně naprogramovali a nastavili můj mikrokontrolér. Při tom jsme zjistili, že jsem patice čipu dal na opačnou stranu cuprextitu. A to jak u zařízení, tak u programátoru. Tímto jsme nalezli hlavní problém mého zařízení.

Přeletoval jsem tedy patici na čip na správnou stranu a vložil do ní již nastavený mikrokontrolér (Obrázek 12). Tuto úpravu jsem udělal rovněž u svého programátoru, pak byl již zcela funkční.

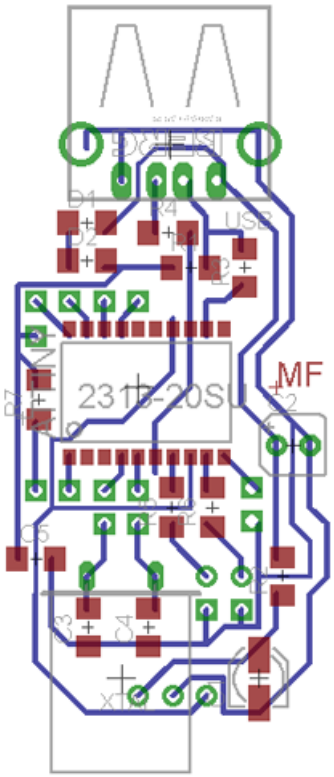
Takto upravené zařízení již po připojení k počítači reagovalo korektně. Rozsvítily se obě LED diody a ovladač k zařízení se nainstaloval bez problémů. Testovací aplikace pana Ing. Česka zařízení našla také a při stisku klávesy dálkového ovladače se zobrazil graf příslušného kódu. Zařízení bylo tedy naprosto funkční a hotové, avšak ne příliš skladné a hezké.

Celková cena programátoru je 142, 90 Kč. (Tabulka 3)

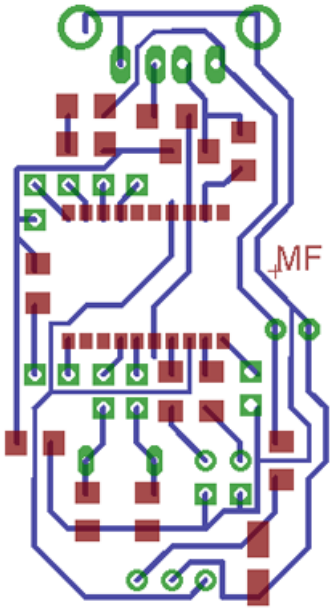
2.5 Verze třetí

Proto jsem se rozhodl pro třetí verzi svého zařízení, ve kterém bych zužitkoval vše, co jsem se do té doby naučil. Bez chyb, kterých jsem se doposud dopustil na předchozích verzích. U třetí verze zařízení jsem se rozhodl pro zcela nový postup výroby a pro využití SMD součástek, kvůli zmenšení celého zařízení. A vzhledem k tomu, že neexistuje žádná patice pro SMD mikrokontrolér (ATTINY 2313-20SU) a přímo se letuje na cuprextit, rozhodl jsem se pro výstupní piny pro připojení programátoru přímo k zařízení bez nutnosti odpojovat samotný mikrokontrolér.

Schéma třetí verze mého zařízení je shodné s druhou verzí (Obrázek 7), jelikož až na přidání výstupních konektorů pro programátor nebylo třeba již funkční schéma nijak měnit. Jediné, co se tedy oproti druhé verzi zařízení změnilo, je typ použitých součástek, a to SMD součástky. Schéma rozvržení součástek jsem vytvořil ve volně dostupném programu Eagle[3] (Obrázek 18, 19).



Obrázek 18 – schéma se součástkami



Obrázek 19 – schéma bez součástek

U třetí verze zařízení jsem použil jiný postup výroby desky tištěného spoje (dále jen DPS). Schéma vytvořené v Eaglu jsem zrcadlově vytiskl na laserové tiskárně na papír. Jako první jsem vyzkoušel obyčejný kancelářský papír, ale jeho vlastnosti nebyly k tvorbě DPS ideální. Barva z laserové tiskárny totiž k obyčejnému papíru příliš přilnula a při jejím přenosu na cuprextit se některé cesty přerušily. Dále jsem vyzkoušel barevné lepící papíry od výrobce Stepa s.r.o., pro tohoto výrobce jsem se rozhodl díky tomu, že přímo na internetových stránkách uvádí, že se papíry hodí pro výrobu DPS.[4]

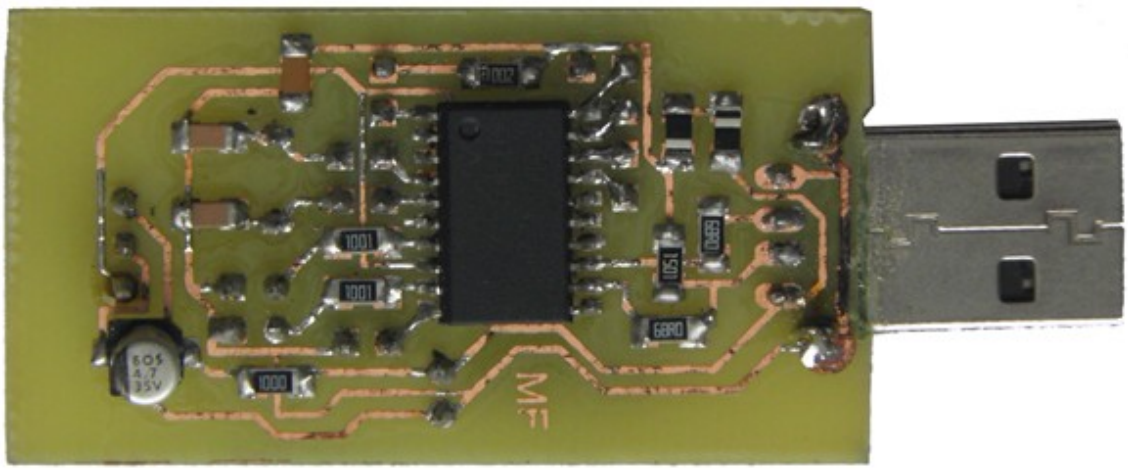
[3] URL: <http://www.elcad.cz/eagle/> [cit. 2011-07-19]
[4] URL: <http://www.stepa.cz/katalog/slozky/vyroba_dps.pdf> [cit. 2011-07-21]

Po vytištění schématu na lepivou stranu barevného papíru jsem nažehlil barvu na cuprextit. Postup nažehlení: vespod je tvrdá podložka, pak vrstva látky, natištěné schéma, cuprextit, obyčejný papír pro ochranu žehlící plochy a žehlička. Zhruba minutu se nechá žehlička jen volně ležet, aby se dostatečně a rovnoměrně prohřál cuprextit. Další minutu leží žehlička s přítlakem asi 1kg. Až cuprextit s nažehlenou vrstvou barvy zchladne, vloží se na 5-10 minut do studené vody. Poté už stačí pouze opatrně sloupnout prázdný papír a nechat uschnout a případně nesmývatelným značkovacím fixem opravit vzniklé chyby (Obrázek 20).

Cuprextit s nažehlenou vrstvou barvy se pak vloží do leptacího roztoku (Chlorid železitý – $FeCl_3$), ve kterém se vyleptá nepokrytá měď. Leptání trvá 5-20 minut v závislosti na mnoha faktorech. Leptání probíhá rychleji v zahřátém roztoku (stačí nechat lahvičku s roztokem pod proudem horké vody). Další faktor urychlující leptání je míchání. Ve statickém roztoku se leptá pomaleji, proto je dobré s nádobkou, ve které leptání probíhá, občas zamíchat. Asi nejdůležitější faktor, který ovlivňuje leptání, je čistost roztoku. Roztok lze totiž používat opakovaně. Po vyleptání nepotřebné mědi jsem téměř hotové DPS omyl horkou vodou. (Obrázek 21)

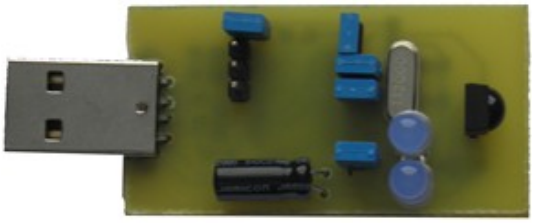
Dále jsem očistil zbytky barvy, která zůstala na mědi i po umytí vodou. K odstranění toneru z laserové tiskárny je vhodné použít Nitroředidlo C6000 (Obrázek 22). Pak je nutné vyvrtat do cuprextitu potřebné dírky pro součástky s nožičkami. K tomu jsem použil vrtáček o průměru 1 mm. Takto navrtané DPS jsem ještě ošetřil asi dvaceti procentním roztokem kalafuny v Nitroředidle C6000 (Obrázek 23). Roztok má zlatavou barvu a i po zaschnutí je stále lehce lepkavý. Slouží k ochraně mědi před oxidací vnějšími vlivy a zároveň usnadňuje pájení.

Takto připravené DPS jsem osadil součástkami (Obrázek 24). Poté jsem k příslušným pinům (Obrázek 25) připojil programátor a připojil přes LPT port k počítači. Tam jsem díky PonyProgu2000 nahrál do mikrokontroléru obslužný program a nastavil pojistky na správné hodnoty. Pak už stačil pouze odpojit programátor a místo připojených konektorů vložit na piny jumpery, aby se mikrokontrolér spojil se zbytkem zařízení (Obrázek 26). A finální verze mého zařízení byla hotová a naprosto funkční.



Obrázek 24 – třetí verze zařízení osazená SMD součástkami

Obrázek 25 – viditelné piny k připojení programátoru



Obrázek 26 – hotová třetí verze zařízení s nasazenými jumpery

Celková cena třetí verze zařízení je 181, 90 Kč.

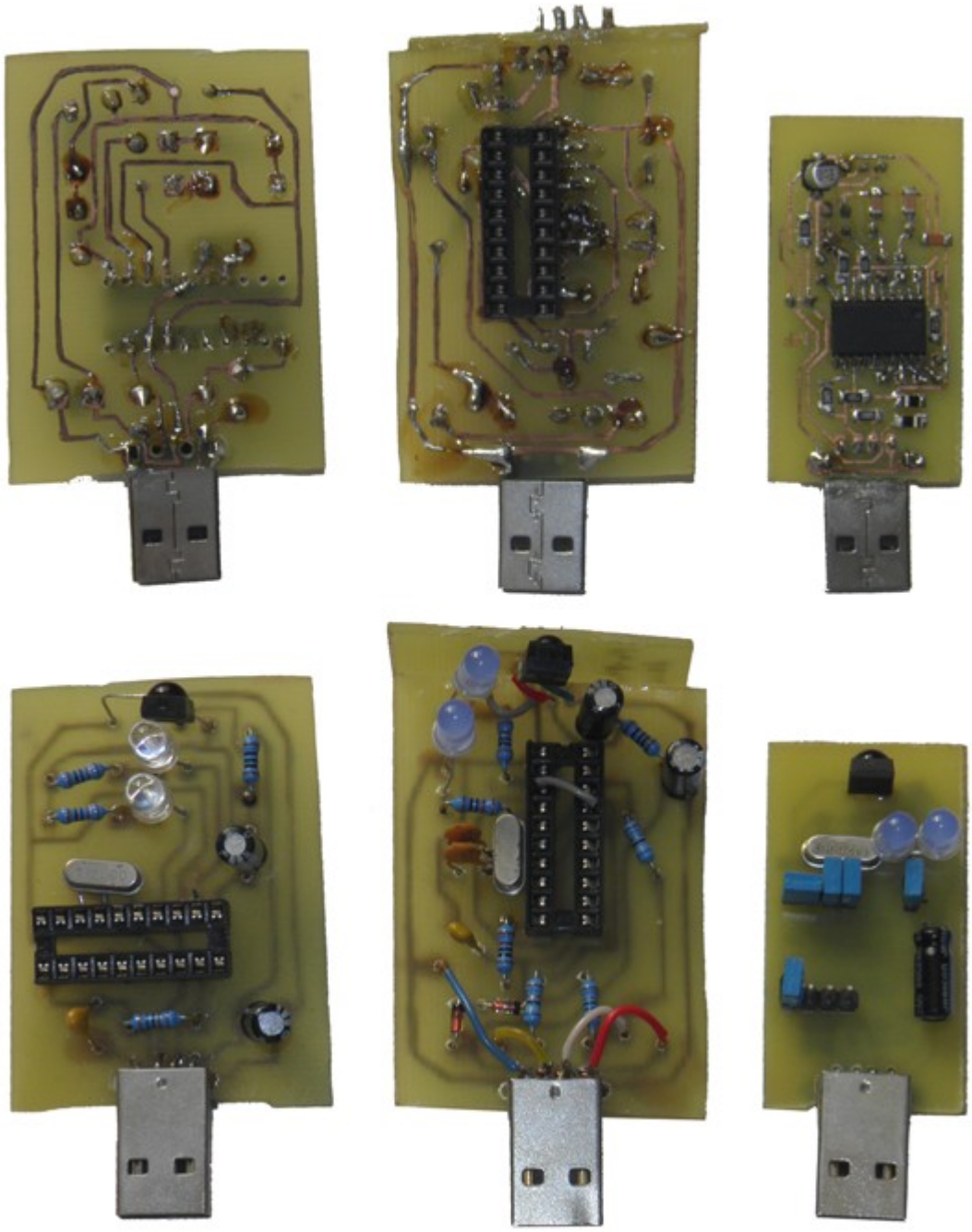
2.6 Závěrečné shrnutí konstrukcí

Zhotovil jsem tři verze zařízení schopného číst infra červený kód z dálkového ovladače a převést ho do počítače přes v dnešní době nejpoužívanější a nejběžnější rozhraní, a to USB 2.0. (Obrázek 27)

První verze zařízení sice v podobě, v jaké se nyní nachází, není funkční, avšak jednoduché otočení patice na mikrokontrolér na opačnou stranu by problém odstranilo. Zařízení je však zbytečně velké a nepraktické kvůli použití běžných součástek. Navíc jeho cena je zbytečně vysoká.

Druhá verze zařízení oproti první funguje bez problémů, patice je totiž již otočena na správnou stranu. Ale ostatní problémy první verze neodstraňuje. Naopak je ještě rozměrnější a dražší než první verze. Proto také není ideální.

Třetí verze zařízení je naopak mnohem menší než předchozí dvě verze, díky použití mikro součástek. Zároveň má k dispozici piny pro přímý přístup k mikrokontroléru bez nutnosti demontáže. Dokonce i jeho cena je značně nižší, než je tomu u předchozích dvou verzí. Tato verze zařízení je konečná a v mých očích je ideálním výběrem obvodu umožňujícím příjem IR signálu z dálkového ovladače.

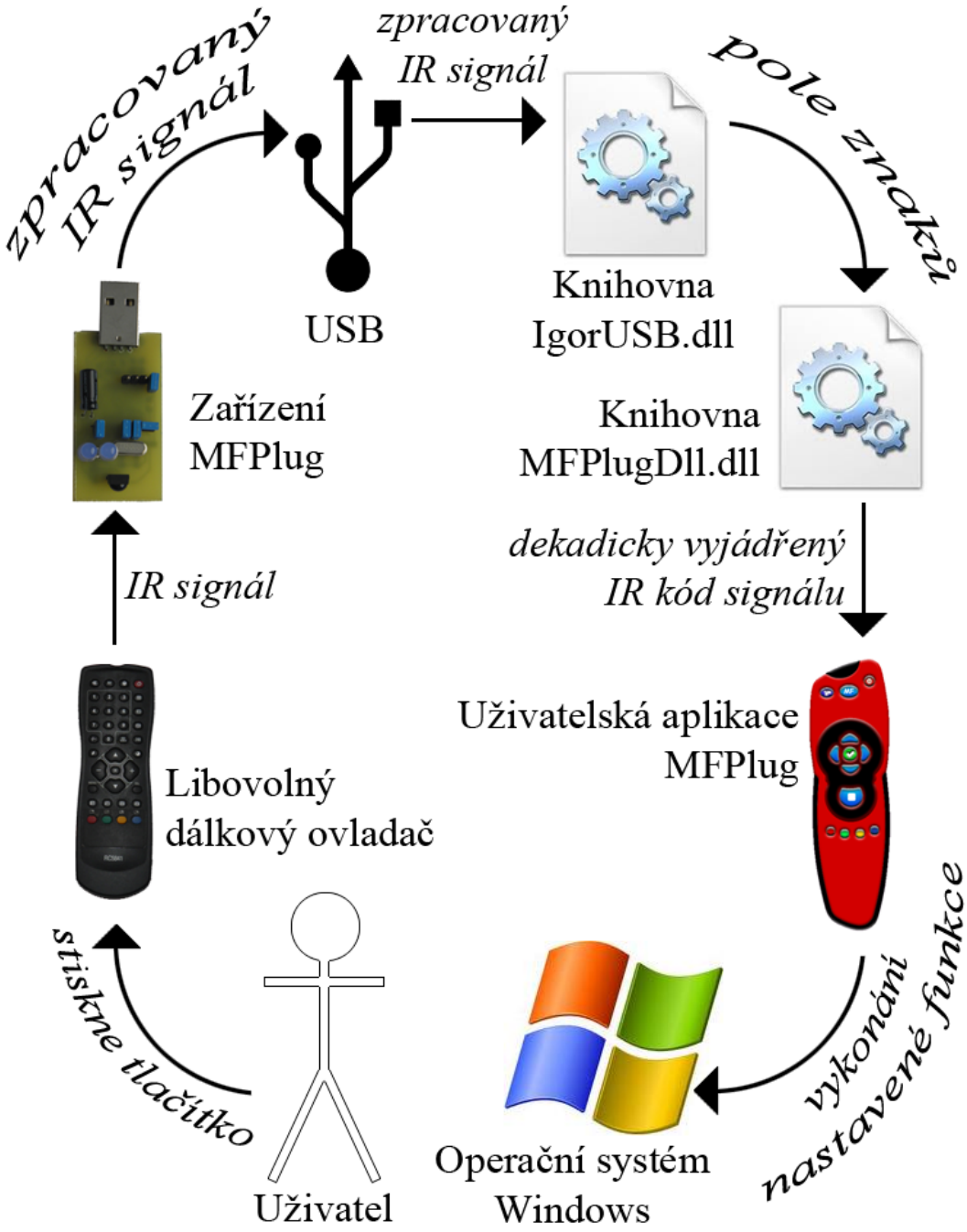


Obrázek 27 – porovnání všech tří verzí dohromady

3. Návrh a implementace uživatelské aplikace

Uživatelskou aplikaci jsem se rozhodl implementovat v jazyce C#. Důvodů bylo víc, mezi hlavní určitě patří, že je mi při tvorbě GUI mnohem bližší než např. Java. Také s ním mám při tvorbě složitějších aplikací víc zkušeností.

Než jsem přistoupil k samotné implementaci uživatelské aplikace, musel jsem si rozmyslet a uvědomit si cestu, jakou signál z dálkového ovladače musí urazit k ovlivnění chování operačního systému. Nejprve při stisku tlačítka dálkového ovladače signál přijme připojené zařízení a firmware ho zpracuje. Zpracovaný signál putuje přes rozhraní USB, pak jej přijme knihovna IgorUSB.dll, napsaná panem Ing. Českem. Tato knihovna je napsána v jazyce C a je volně dostupná. Její volná verze však při zahájení a ukončení spojení zobrazí upozornění, že se jedná o volnou verzi knihovny. S touto knihovnou jsem neměl žádný problém a byla ideální volbou při dekódování kódu. Její přímé použití v koncové uživatelské aplikaci není však díky vnitřním bezpečnostním opatřením operačního systému možné. Proto bylo nutné napsat vlastní knihovnu, která bude prostředníkem mezi IgorUSB.dll a uživatelskou aplikací. O samotné implementaci mé knihovny, která slouží právě jako prostředník, se více zmíním později. Signál z ovladače tedy putuje přes zařízení do USB, dále přes IgorUSB.dll do mé knihovny a z ní již jako celé číslo v dekadickém formátu přímo do uživatelské aplikace, která signál zpracuje a provede nastavenou operaci k dané klávese.



Obrázek 28 - diagram cesty signálu z dálkového ovladače

3.1 Implementace pomocné knihovny

První věc, kterou jsem potřeboval udělat, bylo vytvoření pomocné knihovny. Tato knihovna slouží jako prostředník mezi IgorUSB.dll a aplikační vrstvou uživatelské aplikace. Knihovna je implementována v jazyce C++. Knihovnu jsem pojmenoval MFPlugDll.dll a

vytvořil jsem ji jako samostatný projekt. Vytvoření samostatné knihovny totiž může pomoct ostatním vývojářům, kteří by měli zájem napsat si vlastní aplikaci ke svému zařízení. Proto je napsána v angličtině. Toto zařízení by však muselo být, stejně jako mé, závislé na IgorUSB.dll, tedy na práci pana Ing. Česka.

Knihovna MFPlugDll.dll obsahuje 6 externích metod pro práci a správu IgorUSB.dll, a tím i zařízení. Dále bych se rád věnoval jednotlivým externím metodám. [5]

ConnectIgorUsb – tato metoda vytvoří instanci spojení s knihovnou IgorUSB.dll, tím vyvolá zprávu o volné verzi této knihovny.

DisconnectIgorUsb – metoda sloužící pouze k propuštění instance spojení. Také vyvolá zprávu o verzi knihovny IgorUSB.dll.

DoGetInfraCode – asi nejdůležitější metoda mé knihovny. Slouží k příjmu infra červeného kódu, jeho dekódování a převodu na celé dekadické číslo. Toto číslo je vráceno jako výstupní parametr. Dekódování signálu probíhá tak, že přijatý signál z IgorUSB.dll je nejprve přijat jako pole neoznačených znaků (unsigned char array) s maximální délkou 256 znaků. Toto pole obsahuje znaky, jež jsou po převedení na celé číslo rovny délce daného impulsu nuly nebo jedničky. Pro jejich rozpoznání je nutné tato čísla porovnat s minimem a maximem v přijatém poli, avšak s použitím odchylky. Délka jednotlivých impulsů není totiž vždy stejná, mění se však jen lehce. Takto přijaté pole může vypadat následovně.

index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Kód 1	9	11	20	11	10	10	11	10	11	10	10	10	11	10	11	10	10	21	10	11	21	10	10
Kód 2	9	11	21	10	10	10	11	10	11	10	11	10	10	10	11	10	10	21	11	10	21	10	10

Tabulka 5 – příklad kódu přijatého signálu z dálkového ovladače

Několik prvních znaků je pouze zaváděcí stopou a mohou se měnit při opakovaném stisku klávesy. Pro výslednou hodnotu jsou neužitečné. Proto dekódování probíhá až od znaku na indexu 5. V dll je toto reprezentováno konstantou „počáteční index“. Hodnoty blízké minimu v poli označují nulový bit a hodnoty blízké maximu označují jedničkový bit. K určení blízkosti slouží konstanta „odchylka“, její hodnota je tři. Zkoušel jsem mnoho různých dálkových

[5] MAYO, J.; *C# 3.0 Unleashed With the .NET Framework 3.5*. 2. vyd. USA: Pearson Education, Inc., 2008. str. 864

ovladačů a hodnoty se nikdy nelišily o více než tři, spíše však o hodnotu jedna až dvě. Kód 1 i 2 z tabulky tedy odpovídají hodnotě $(0000000000000100100)_2 = (36)_{10}$. Oba kódy jsou reprezentací jediné klávesy, i když je patrné, že pole nejsou ekvivalentní bez použití odchylky. Metoda tedy pro tyto konkrétní kódy ve výstupním parametru vrátí hodnotu 36.[6]

DoSetDataPortDirection – tato metoda slouží k nastavení směru přenosu na pinech mikrokontroléru. Tím mimo jiné ovlivňuje i svítivost LED diod.

DoSetOutDataPort – metoda nastavující stav výstupních pinů, nebo umožňující nastavení pull - up rezistorů mikrokontroléru. Slouží např. k vypnutí a zapnutí LED diod.

IsDeviceReady – touto metodou se ověřuje, zda je zařízení připojeno k počítači. Knihovna zkouší zapsat hodnotu do vnitřní EEPROM paměti mikrokontroléru. Pokud se zápis podaří, zařízení je připojeno.

Knihovna IgorUSB.dll obsahuje navíc ještě několik metod k obsluze připojeného zařízení. Jsou to metody, které umožňují například číst nebo zapisovat do vnitřní EEPROM paměti, metody ovlivňující rychlost mikrokontroléru na lince RS232 a další. Tyto metody jsem však u uživatelské aplikace použít nechtěl, proto jsem je do své knihovny nezakomponoval.

3.2 Implementace uživatelské aplikace

Uživatelskou aplikaci jsem pojmenoval MFPlug. Implementována je v jazyce C# a její Solution obsahuje dva projekty. Asociace jednotlivých tříd těchto projektů je patrná na třídním diagramu (Obrázek 29). První projekt, MFPlug je Windows Form Application[7], který obsahuje potřebné formuláře k chodu programu. Druhý projekt je pouze Class Library, knihovna obsahující třídy nutné pro aplikaci a správu operačního systému při stisku tlačítka ovladače a provedení příslušné funkce.

3.2.1 MFPlug - Windows Form Application

Tento projekt tvoří hlavní část uživatelské aplikace. Obsahuje formuláře zobrazující se uživateli. Formulář představuje okna nebo dialogové boxy, které tvoří uživatelské rozhraní aplikace. Projekt jsem začal tvořit jako klasickou oknovou aplikaci. V průběhu práce jsem však změnil názor. Chtěl jsem totiž, aby má aplikace byla něco víc než jen běžný formulář. Proto

[6] TROELSEN, A.; *C# a .NET 2.0 Profesionálně.* 1. vyd. Brno: ZONER software, s.r.o., 2006. str. 136
[7] TROELSEN, A.; *C# a .NET 2.0 Profesionálně.* 1. vyd. Brno: ZONER software, s.r.o., 2006. str. 753

jsem začal s tvořením grafiky. Hlavní motiv mi byl jasný hned, dělal jsem přece dálkové ovládání počítače. A tak jsem se rozhodl pro jednoduchý dálkový ovladač. Nakreslil jsem pomocí grafického programu obrázek ovladače, jednotlivých tlačítek, a to aktivních i neaktivních. Obrázek aktivního tlačítka se zobrazí, když se kurzor dostane nad dané tlačítko. Celý ovladač jsem vytvořil hned ve čtyřech různých barvách. Podle barevných tlačítek, která jsou na většině běžných dálkových ovladačů. Uživatel tak má možnost volby z červené, zelené, žluté nebo modré barvy ovladače. Dále jsem začal s tvorbou dalších formulářů, které byly potřeba k chodu aplikace. Teď popíšu jednotlivé formuláře mého programu a budu se více věnovat každému z nich jednotlivě.

F_Hlavní - hlavní formulář, který se zobrazí při spuštění aplikace. Jeho grafická podoba je právě dálkový ovladač. Slouží jako centrum celé aplikace a je jediný, který lze vidět na hlavním panelu, nebo i při cyklování oken přes klávesy alt + tab. Zobrazuje také ikonku v Systém tray, která slouží třeba pro ovládání minimalizovaného programu.

Metody hlavního formuláře obsluhují například pohyb programu po monitoru, ukončení celé aplikace nebo zobrazení ostatních formulářů. Při spuštění aplikace se v tomto formu zjistí, zda je k počítači připojeno zařízení a zároveň je spuštěno paralelní vlákno, které za běhu programu testuje, zda bylo zařízení odpojeno nebo připojeno k počítači. Tento formulář také řídí nastavení barev, při změně se nastaví nová barva nejen na tomto formuláři, ale také na všech ostatních prvcích programu. Hlavní formulář obsahuje konstanty, týkající se pozice ostatních formulářů, nutné pro jejich správné zobrazování a posunování. Obsluhuje také samotné čtení kódu z ovladače, resp. z knihovny MFPlugDll.dll. Pokud je čtení aktivní, kód je zde zpracován paralelním vláknem, spuštěným právě při aktivaci čtení. Poté je volána funkce odpovídající hodnotě přečteného kódu.

F_NastaveníFunkceKláves - pouze pomocný formulář, sloužící jako pozadí pro formulář F_Funkce, o kterém se zmíním později. Tento formulář se zobrazí při stisku tlačítka na formuláři F_Hlavní. Zobrazení probíhá animací, ta je sice implementována ve formuláři F_Hlavní, týká se však tohoto formuláře, proto ji popisuji zde. Animace tedy probíhá posunem lokace a zvětšováním velikosti tohoto formuláře až do maximální šířky. Aby byla animace dokonalá, musel jsem rozdělit daný formulář na šest částí, a to proto že jeho šířka je mnohem větší než šířka hlavního formuláře. Pokud bych tedy zobrazil hned celý obrázek formuláře, byl by vidět zpoza hlavního, a to by nebylo dobré. Algoritmus roztažení nastavení tedy funguje tak, že se postupně posunuje celý formulář směrem doprava a zároveň se při dosažení určité lokace změní obrázek na jeho pozadí za obrázek větší. Při každém kroku animace vlákno, které

roztažení obstarává, vyčkává 15 milisekund, aby se docílilo hladšího průběhu animace. Algoritmus animace skrývání je stejný, avšak obrácený. Při úplném zobrazení se zobrazí formulář F_Funkce.

F_Funkce - tento formulář slouží k nastavení funkcí. Obsahuje také statické metody, které reprezentují jednotlivé funkce a konstantní proměnné k nim potřebné. Funkcí je možné přidat až sto a dovolují nastavení několika různých dálkových ovladačů zároveň. Nastavení funkce probíhá tak, že se nejprve přečte kód tlačítka dálkového ovladače, kterému se má přiřadit nějaká funkce. Tento kód se uloží do matice ve statické třídě IsolatedStorage (dále jen IS). Index matice je číslo nastavované funkce a pozice na daném indexu je konstanta ze třídy IS „pozice kódu klávesy“. Čtení klávesy proběhne v paralelním vláknu, aby se nezbrzdil chod programu. K tomuto čtení je potřeba mít připojeno zařízení. To je zajištěno metodou, která testuje před každým čtením kódu, zda je zařízení připojeno k počítači. Když je načten kód, ComboBox, jenž obsahuje seznam funkcí, přejde do stavu Enabled = true. Tímto je uživateli umožněn výběr funkce. Ten probíhá tak, že se vybere funkce ze seznamu ComboBoxu. Při změně vybraného indexu je vyvolána příslušná událost. Vybraný index funkce se uloží na stejný index v matici, tedy pořadové číslo dané funkce. Pozice na tomto indexu je rovna konstantě z IS „pozice kódu funkce“. Dále je zobrazen formulář F_NovéJménoFunkce, kde uživatel zadá název nastavené funkce. Pro tento název jsem se rozhodl proto, aby měl uživatel větší přehled o nastavených funkcích. Nový název je rovněž uložen v IS do příslušného pole na pozici, jež odpovídá pořadí dané funkce. Při výběru funkce simulující stisk klávesy klávesnice je uložen index dané klávesy do IS. Při výběru ostatní klávesy je do IS uložen znak této klávesy i s vybranými modifikátory.

F_NovéJménoFunkce - pouze malý formulář, který obsahuje TextBox a tlačítko k potvrzení. Do TextBoxu se vloží nový název funkce, jež je právě nastavena.

F_VýběrKlávesy - tento formulář se zobrazí při výběru funkce simulující stisknutí klávesy klávesnice. Slouží právě k výběru klávesy, jež má být stisknuta, je-li funkce zavolána. Formulář obsahuje ComboBox se seznamem kláves, které mohou být stisknuty. Jsou zde například klávesy Enter, Backspace, Mezerník, Esc a další. Vybraná možnost, tedy index vybrané klávesy z ComboBoxu je uložen v IS do příslušného pole s indexem, jenž odpovídá pořadovému číslu funkce a konstantě „pozice“ Je zde také možnost Ostatní klávesy. Ta zobrazí formulář F_ČteníKlávesy.

F_ČteníKlávesy - formulář sloužící při volbě ostatní klávesy, tedy jiné, než jsou přímo na výběr ze seznamu ve formuláři výše. Tento formulář čeká, až uživatel stiskne klávesu, kterou chce přiřadit funkci. Při stisku klávesy je volána událost OnKeyPress, kterou jsem upravil do takové podoby, že čte klávesy písmen a čísel, tedy znaky a-z, A-Z a 0-9. Stisknutý znak je zobrazen v Labelu a je možné k němu přidat i další klávesu, takzvaný modifikátor (Ctrl, Alt nebo Shift). Toto slouží k vyvolání téměř libovolné klávesové zkratky.

F_oProgramu - zobrazuje informace o programu MFPlug a také možnosti nastavení samotného programu. Jeho zobrazení, tedy vysunutí z boční strany hlavního formuláře je stejně jako u nastavení funkce kláves zajištěno animací v hlavním formuláři. Tato animace je ale jednodušší než předešlá. Algoritmus animace vysunutí se skládá pouze z posunu lokace a změny šířky tohoto formuláře. Změna šířky totiž naprosto stačí k tomu, aby zobrazovaný formulář nevyčníval zpod hlavního formuláře. Rostoucí šířka totiž roztahuje formulář směrem doprava a při vysouvání formuláře doleva je dostačující pouze posunout zároveň její lokaci směrem doleva. Při plném vysunutí se zobrazí všechny prvky tohoto formuláře. Ty jsou totiž během animace skryty pro dosažení kvalitnější animace. Animace zasunutí je opět pouze převrácený algoritmus vysunutí. K prvkům ovlivňující zařízení patří trojice RadioButtonů ovlivňující svítivost LED diod zařízení. Svítivost je ovládána přes metody knihovny MFPlugDll.dll, o nichž jsem se již zmiňoval. Dále dvojice RadioButtonů, které umožňují nastavení jazyka uživatelské aplikace. Jazyky jsou k dispozici dva, a to český jazyk a anglický jazyk. Nakonec je zde CheckBox, ovlivňující zda je aplikace vždy navrchu, tedy hodnotu formulářů TopMost. Všechny nastavené hodnoty jsou při změně uloženy do IS, aby bylo možné při znovu spuštění programu navázat na nastavené hodnoty uživatele bez jeho zásahu.

F_Nápověda - posledním formulářem je nápověda. Nápověda obsahuje detailní popis všech prvků mého programu. K implementaci zde není moc co dodávat. K popisu jednotlivých tlačítek slouží Labely, které obsahují informace o tlačítkách. Obsah Labelů s popisy se mění v závislosti na vybraném jazyce.

3.2.2 MFPlug_Classes - Class Library

Druhý projekt uživatelské aplikace. Knihovna obsahující třídy nutné k obsluze operačního systému, uložení nastavení programu a funkcí. Také ke spojení se s knihovnou MFPlugDll.dll, čtení a zpracování přijatého kódu. Všechny třídy, které tato knihovna obsahuje,

jsou statické.[8] Nejsou tedy nikde instalovány a všechny jejich metody a parametry jsou buď statické, nebo konstanty. Ty se totiž díky faktu, že jsou neměnné, chovají jako statické parametry. Výhodou statické metody je to, že již kompilátor ověřuje, zda neexistuje žádné vytvoření instance této třídy. Tyto třídy jsou také zapečetěné, a díky tomu se z nich nedá dědit. Dále se budu podrobněji věnovat jednotlivým třídám knihovny.

MFPlugDll - tato třída obsahuje pouze deklaraci konstant a externích metod knihovny MFPlugDll.dll. Ty jsou volány přímo z knihovny pomocí DllImportu. Třída MFPlugDll je pouze interní, to znamená, že použít ji mohou pouze třídy v knihovně MFPlug_Classes. Konkrétně jsou metody této třídy použity ve třídě InfraCode.

InfraCode - třída obsluhující práci se zařízením. Příjímáním a zpracováním infra červeného kódu ze zařízení, resp. knihoven MFPlugDll.dll a IgorUSB.dll. Metody importované ve třídě MFPlugDll jsou zde volány a v případě potřeby i upraveny. Zároveň jsou z této třídy dál k dispozici v již použitelné podobě. Například metoda IsDeviceReady vrací pouze booleovskou hodnotu True nebo False, jež reprezentuje stav připojení zařízení. Ne jen číslo, rovnající se případné chybové hlášce, jak je tomu u třídy MFPlugDll, což je pro další využití praktičtější. Také metoda ReadInfraCode je upravena, aby vracela přímo přečtený kód a ne chybovou hlášku s informací o připojení zařízení. Dále je zde metoda k upravení svítivosti diod podle hodnoty parametru, jež je vložen. Jsou zde i konstanty nutné pro správu této třídy a statické proměnné, udávající zda je aktivní čtení kódu.

IsolatedStorage - než se začnu věnovat mé třídě, uvedu obecné výhody IsolatedStorage, tedy izolovaného úložiště dat (dále jen IS). Když aplikace ukládá data do souboru, musí být název souboru a místo uložení pečlivě vybírány, aby byla minimalizována možnost, že úložiště bude známo jiné aplikaci, a tudíž náchylné k poškození. IS je mechanismus pro ukládání dat poskytující izolaci a bezpečnost definováním standardizovaných způsobů asociace kódu s uloženými daty. Standardizace také poskytuje další výhody. Správci mohou používat nástroje, které jsou navrženy pro nastavení kapacity IS, nastavení zásad zabezpečení a odstraňování nepoužitých dat. S IS již kód dále nepotřebuje jedinečné cesty pro určení bezpečných umístění v souborovém systému a data jsou chráněna před jinými aplikacemi, které mají přístup k IS. Pevně zakódovaná informace, která označuje, kde je umístěna oblast úložiště aplikace, není potřebná. S IS jsou data vždy izolována konkrétním uživatelem a sestavení (anglicky assembly). V mém případě navíc i doménou. Tímto je zajištěn víceuživatelský přístup

[8] TROELSEN, A.; *C# a .NET 2.0 Profesionálně.* 1. vyd. Brno: ZONER software, s.r.o., 2006. str. 145

k aplikaci, jelikož každému uživateli bude vytvořen samostatný IS se souborem obsahujícím nastavením programu a jednotlivých funkcí. Při použití IS tedy aplikace ukládá data do unikátního prostoru. Tento prostor je abstraktní, není to konkrétní úložiště. Skládá se z jednoho nebo více souborů nazývaných anglicky Stores. Tyto soubory obsahují aktuální místo adresáře, ve kterém jsou data uložena. Data uložená v IS mohou být jakéhokoliv typu. V mém případě je to XML soubor[9], obsahující vše potřebné k obnově nastavení programu a znovu vypsání nastavených funkcí při opětovném spuštění aplikace.[10]

Teď tedy zpět k mé třídě. Jak jsem již zmínil, můj IS využívá XML soubor, v němž jsou zapsány všechny potřebné informace, tedy hodnoty statických proměnných této třídy. Jsou jimi uživatelem vybraná barva a jazyk, ale také lokace hlavního formuláře. Dále nastavená úroveň svítivosti diod, informace o tom, zda má být aplikace zobrazována vždy navrchu a v neposlední řadě také informace o všech nastavených funkcích, včetně jejich celkového počtu. U funkcí se zaznamenává pořadové číslo funkce, vložené jméno funkce, hodnota infra červeného kódu tlačítka dálkového ovladače v dekadickém tvaru, číslo funkce, tedy index vybrané funkce ze seznamu. Při výběru funkce simulující stisk klávesy klávesnice je uložen také index reprezentující vybranou klávesu. U výběru ostatní klávesy je zde uloženo číslo znaku této klávesy a zda byly vybrány nějaké modifikátory (Ctrl, Alt nebo Shift).

Ukázka výsledného XML souboru se nalézá v příloze 10/1.

Třída IsolatedStorage dále obsahuje metody potřebné k uložení XML a opětovného načtení z něj zpět do proměnných. Také obsahuje metody potřebné k rozpoznání čísla funkce podle kódu stisknutého tlačítka. U funkce simulující stisk klávesy klávesnice hledá v matici příslušnou klávesu a informace k ní (zadané modifikátory apod.).

MouseControl - třída, která slouží k simulaci stisknutí tlačítka myši. Tuto třídu používá formulář F_Funkce pro vykonávání funkcí, jež souvisí právě se simulací tlačítek myši. Využívá k tomu importovanou metodu mouse_event z knihovny user32.dll. Tato metoda má čtyři důležité parametry, jsou to typ stisknutého nebo uvolněného tlačítka, parametry x a y, tedy absolutní pozice kurzoru na ose x resp. y. A nakonec je to hodnota posuvu při točení kolečkem.

KeyboardControl - třída sloužící k simulaci funkcí klávesnice. Tedy stisknutí kláves, a to nejen těch běžných, ale i speciálních kláves, které jsou pouze na multimediálních

---

[9] BAYER, J.; *C# 2005 Velká kniha řešení.* 1. vyd. Brno: Computer Press, a.s., 2007. str. 343

[10] URL: <http://msdn.microsoft.com/cs-cz/library/3ak841sy(v=VS.100).aspx> [cit. 2011-07-22]

klávesnicích. Jsou zde mimo jiné klávesy ovládající chod multimediálního přehrávače Windows Media Player, jenž je součástí operačního systému Windows, dále také například klávesy ovládající internetový prohlížeč. Třída KeyboardControl opět používá importovanou metodu z knihovny user32.dll, a to metodu keybd_event. Mezi důležité parametry této metody patří virtuální kód stisknuté klávesy a číslo, které udává, zda byla klávesa stisknuta nebo propuštěna. Všechny použitelné virtuální kódy třída obsahuje ve formě veřejných konstant.

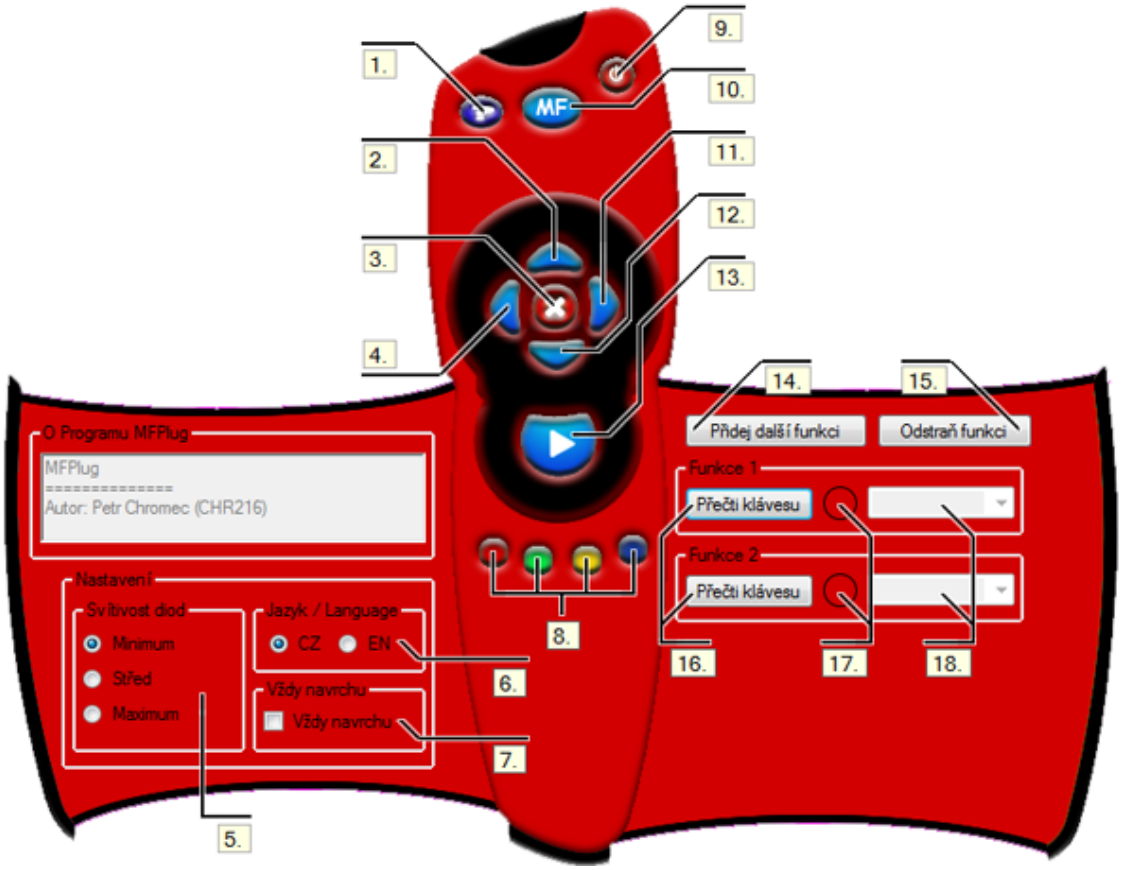
WinampControl - tato třída ovládá přímo program Winamp. Slouží k tomu dvě metody, ty jsou opět importovány z knihovny user32.dll. Importované metody jsou FindWindow a SendMessageA. FindWindow předá ukazatel na okno, které odpovídá zadanému názvu v parametru. Metoda SendMessageA pošle zprávu oknu, které vrátí metoda FindWindow, tedy v mém případě právě programu Winamp. Tato zpráva obsahuje kód, aby Winamp rozpoznal, že zpráva obsahuje příkaz pro něj. Dále obsahuje kód samotného příkazu. Kódy příkazů jsou ve třídě uloženy jako veřejné konstanty. Jejich hodnoty jsem nalezl ve volně dostupném SDK programu Winamp.[11]

[11] URL: <http://forums.winamp.com/showthread.php?s=&threadid=168643> [cit. 2011-07-25]

4. Uživatelská příručka programu MFPlug

V této kapitole se budu věnovat popisu programu z uživatelského hlediska, tedy jak se s programem MFPlug vlastně pracuje, jak vypadá a jak funguje.

Ze všeho nejdříve bych rád ukázal okno nápovědy (Obrázek 30), to je součástí samotného programu. Nápověda ukazuje, k čemu každé tlačítko slouží. Všem tlačítkům jsem přiřadil číslo, kvůli jednoduššímu a přehlednějšímu rozpoznání a popisu.



Obrázek 30 – nápověda

1. Nápověda
2. Ulož změny
3. Stav zařízení
4. Zasune nastavení kláves
5. Nastavení svítivosti diod
6. Nastavení jazyka
7. Zobrazení programu nad všemi ostatními
8. Změna barvy programu

9. Zavřít
10. O programu MFPlug
11. Vysune nastavení kláves
12. Minimalizovat
13. Start / Stop čtení kódu
14. Přidej další funkci
15. Odeber funkci
16. Přečti klávesu
17. Ukazatel stavu čtení kódu
18. Výběr funkce

4.1 Hlavní okno

V této části se zaměřím na hlavní okno programu MFPlug. (Obrázek 31) Sestává z třinácti tlačítek a je hlavní částí uživatelské aplikace. Z této centrální části se řídí hlavní funkce programu, jsou zde také tlačítka, která otevírají okna pro nastavování nebo nápovědu. Hlavní okno je také jedinou částí, kterou lze pomocí myši pohybovat. Na tomto pohybu však závisí i ostatní okna programu. A teď už k samotným tlačítkům.

1. **Nápověda** - zobrazí okno nápovědy. (Obrázek 30) Toto okno obsahuje popis všech tlačítek programu.

2. **Ulož změny -** uloží všechny změny provedené v programu. Nastavenou úroveň svítivosti diod na zařízení, vybraný jazyk programu, zda má být program zobrazen vždy navrchu, vybranou barvu, aktuální pozici hlavního okna, celkový počet funkcí a nastavení všech funkcí.

3. **Stav zařízení** – ukáže stav připojení zařízení. Ikonka tohoto tlačítka se mění v závislosti na tom, zda je zařízení připojeno k USB portu počítače, či nikoliv. (Obrázek 32)

4. **Zasune nastavení kláves -** tlačítko slouží k zasunutí okna, kde se nastavují jednotlivé funkce. Tedy okna na pravé straně hlavního panelu programu.

8. **Změna barvy programu** - změní barvu celého programu i jeho částí na barvu stisknutého tlačítka.

9. **Zavřít** - vypne program. Při změně od posledního uložení se při vypnutí programu bude ptát, zda má tyto změny uložit.

10. **O programu MFPlug** - vysune levé boční okno. Toto okno obsahuje základní prvky nastavení uživatelské aplikace a také informace o programu.

11. **Vysune nastavení kláves** - vysune pravé boční okno, kde je možné přidávat, odebírat a také nastavovat jednotlivé funkce k tlačítkům dálkových ovladačů.

12. **Minimalizovat** - minimalizuje program do Systray, oblasti hodin ve Windows. V Systray je zobrazena ikonka programu. Při levém kliknutí se znovu maximalizuje celý program. Pravé kliknutí zobrazí nabídku. (Obrázek 33)

13. **Start / Stop čtení kódu** - toto tlačítko slouží ke spuštění čtení signálu z dálkového ovladače a vykonávání nastavených funkcí. Při jeho stisknutí je zároveň testováno, zda je zařízení připojeno k počítači, bez toho totiž není možné čtení spustit. Signál je zařízením přečten, program poté rozpozná kód tlačítka ovladače a provede nastavenou funkci. Čtení kódu je možno také spustit z nabídky zobrazené při pravém kliknutí na ikonku programu v Systray Windows. (Obrázek 33)

Obrázek 31 – hlavní okno programu MFPlug (s připojeným zařízením)

4.2 Okno nastavení - levé boční okno

Jedná se o boční okno programu, vysunuté při stisknutí tlačítka „O programu MFPlug“, na obrázku nápovědy má toto tlačítko číslo 10. V tomto okně je možné přečíst si základní informace o programu MFPlug a zároveň se zde nastavují základní parametry. Svítivost LED diod připojeného zařízení, jazyk uživatelské aplikace a možnost zobrazení programu nad všemi ostatními, tedy vždy nahoře.

5. **Nastavení svítivosti diod** - nastaví svítivost LED diod připojeného zařízení na vybranou hodnotu. Ty jsou minimální - diody jsou vypnuty; Střed – diody lehce svítí; Maximum – diody svítí maximální svítivostí.

6. **Nastavení jazyka** - vybere jazyk uživatelské aplikace. Na výběr jsou dva jazyky, a to český jazyk a anglický jazyk. Tento výběr ovlivní celou aplikaci, včetně popisů jednotlivých tlačítek i nápovědy.

7. **Zobrazení programu nad všemi ostatními** - při zaškrtnutí tohoto políčka se program bude zobrazovat vždy nahoře, tedy nad všemi ostatními okny.

4.3 Nastavení funkcí kláves - pravé boční okno

Toto boční okno se vysune při stisknutí tlačítka číslo 11. A zasune při stisknutí tlačítka 4. V tomto okně je možné přidávat, odstraňovat a samozřejmě také nastavovat funkce k jednotlivým tlačítkům dálkového ovladače. Každé tlačítko dálkového ovladače může být nastaveno jen jednou. Při opětovném nastavení téhož tlačítka bude volána vždy funkce s nejnižším pořadovým číslem, tedy funkce nejblíže vrchu. Funkce na druhou stranu může být použita na několik tlačítek zároveň.

14. **Přidej další funkci** - přidá funkci, kterou bude možné nastavit. Maximální počet funkcí je 100.

15. **Odeber funkci** - odebere poslední funkci v seznamu, tedy funkci, která je nejníže.

16. **Přečti klávesu** - tlačítko, které při stisknutí zkontroluje, zda je zařízení připojeno k počítači, bez toho není možné čtení spustit. Pokud je zařízení připojeno, spustí čtení kódu dálkového ovladače a program čeká na signál.

17. **Ukazatel stavu čtení kódu** - ukazatel stavu čtení kódu z dálkového ovladače. Jeho barva se mění v závislosti na stavu čtení kódu. Má tři možné barvy, a to červenou, modrou a zelenou. Červená barva značí, že čtení není aktivní a žádný kód není nastaven. Modrá znamená, že čtení právě probíhá a program v této chvíli čeká na stisknutí tlačítka dálkového ovladače, k němuž se bude nastavovat funkce. Zelená barva ukazuje, že kód je úspěšně načten.

18. **Výběr funkce** - k výběru funkce slouží seznam. Ten obsahuje všechny funkce, které umí program MFPlug vykonávat. K výběru funkce stačí pouze vybrat ji kliknutím ze seznamu. Po tomto kliknutí je zobrazeno okno, do kterého se zadá název právě nastavené funkce. (Obrázek 34)

4.3.1 Nastavení funkce k tlačítku dálkového ovladače

Nastavení funkce k tlačítku dálkového ovladače se tedy provádí následovně. (Obrázek 37) Nejprve musíme přidat funkci tlačítkem číslo 15. Poté se stiskne tlačítko číslo 16 „Přečti klávesu“, tím je spuštěno čtení signálu dálkového ovladače, ikonka uprostřed (číslo 17) se zbarví na modro, přejde se tedy do stavu čtení. V této chvíli je nutné stisknout tlačítko dálkového ovladače, kterému chceme přiřadit funkci. Až je požadované tlačítko programem přečteno, ikonka uprostřed se zbarví na zeleno, a tím se zpřístupní výběr funkce (číslo 18). Po výběru funkce ze seznamu se ještě zobrazí okno s řádkem, do kterého se napíše jméno dané funkce. (Obrázek 30.) To slouží pro identifikaci dané funkce a pro snadnější zapamatování. Tímto je celý proces úspěšně ukončen a nastavenou funkci lze vykonávat při spuštění čtení kódu tlačítko číslo 13 „Start / Stop čtení kódu.

4.3.2 Seznam a popis jednotlivých funkcí

Přidání hlasitosti - zvýší hlavní hlasitost operačního systému (dále jen OS).

Ubrání hlasitosti - sníží hlavní hlasitost OS.

Vypnutí / Zapnutí zvuku - vypne nebo zapne zvuk OS.

Překlopit (Alt + Tab) - tato funkce simuluje stisknutí kláves Alt + Tabulátor. Zkratka slouží k přepínání mezi okny bez použití myši.

Funkce myši

< Pohyb myši - pohyb kurzorem myši směrem doleva.

> Pohyb myši - pohyb kurzorem myši směrem doprava.

^ Pohyb myši - pohyb kurzorem myši směrem nahoru.

v **Pohyb myši -** pohyb kurzorem myši směrem dolů.

Levý klik myši - simuluje stisk levého tlačítka myši.

Levý dvoj-klik myši - simuluje dvoj-klik levého tlačítka myši.

Pravý klik myši - simuluje stisk pravého tlačítka myši.

Střední klik myši - simuluje stisk prostředního tlačítka myši.

Kolečko nahoru - simuluje točení kolečkem směrem nahoru.

Kolečko dolu - simuluje točení kolečkem směrem dolů.

Klávesa - tato funkce simuluje stisk klávesy klávesnice. Při výběru funkce se zobrazí okno, obsahující seznam dostupných kláves. (Obrázek 35)

Seznam jednotlivých kláves je následující: Enter, Esc, Backspace, Mezerník, Shift, Ctrl, Levý Alt, Pravý Alt, Tabulátor, F1, F2, F3, F4, F5, F6, F7, F8, F9, F10, F11, F12, Kurzorové šipky, Capslock, Numlock, Print Screen, Home, End, Insert, Delete, Page Up, Page Down a Jiná klávesa.

Při výběru poslední možnosti, tedy Jiná klávesa se zobrazí další okno. (Obrázek 36) V tom je možné nastavit klávesu, která není v předešlém seznam. K výběru však slouží pouze klávesy A-Z a čísla 0-9. K těm je možné přidat i modifikátor, který bude při simulaci stisknutí této klávesy stisknut také. Tímto lze snadno simulovat téměř všechny klávesové zkratky.

Funkce programu Windows Media Player (dále jen WMP)

WMP - Play / Pause – simulace stisknutí tlačítka Play / Pause programu WMP.

WMP - Stop – simulace stisknutí tlačítka Stop programu WMP.

WMP - Další – simulace stisknutí tlačítka Další skladba programu WMP.

WMP - Předchozí – simulace stisknutí tlačítka Předchozí skladba programu WMP.

Funkce internetového prohlížeče

P - Zpět - simulace stisknutí tlačítka Zpět v aktivním internetovém prohlížeči.

P - Vpřed - simulace stisknutí tlačítka Vpřed v aktivním internetovém prohlížeči.

P - Obnovit - simulace stisknutí tlačítka Obnovit v aktivním internetovém prohlížeči.

P - Zastavit - simulace stisknutí tlačítka Zastavit v aktivním internetovém prohlížeči.

P - Domů - simulace stisknutí tlačítka Domů v aktivním internetovém prohlížeči.

Funkce programu Winamp

W - Play - simulace stisknutí tlačítka Play programu Winamp.

W - Pause - simulace stisknutí tlačítka Pause programu Winamp.

W - Stop - simulace stisknutí tlačítka Stop programu Winamp.

W - Další - simulace stisknutí tlačítka Další skladba programu Winamp.

W - Předchozí - simulace stisknutí tlačítka Předchozí skladba programu Winamp.

W - >> - posunutí přehrávané skladby o 5 sekund vpřed.

W - << - posunutí přehrávané skladby o 5 sekund vzad.

W - Přidat hlasitost - zvýšení hlasitosti programu Winamp.

W - Ubrat hlasitost - snížení hlasitosti programu Winamp.

Změna barvy na červeno - změní barvu programu MFPlug na červenou.

Změna barvy na zeleno - změní barvu programu MFPlug na zelenou.

Změna barvy na žluto - změní barvu programu MFPlug na žlutou.

Změna barvy na modro - změní barvu programu MFPlug na modrou.

5. Testování

V této kapitole se zaměřím na testování a kompatibilitu zařízení, klientské aplikace, jejích funkcí, ale také na dálkové ovladače. Testování proběhlo na několika počítačích s různými operačními systémy. Omezil jsem se však pouze na operační systémy Windows a to 32 bitové, protože pro ty jsem svůj program a zařízení koncipoval. Konkrétně jsem vše testoval na pěti různých počítačích. Dva z nich byly klasické stolní počítače a tři byly notebooky. Následující tabulka ukazuje použité operační systémy těchto platforem.

Počítač	**Operační systém (32 bitové)**
Stolní PC č. 1	Windows XP s SP3
Stolní PC č. 2	Windows XP s SP3
Notebook č. 1	Windows Vista Home Premium
Notebook č. 2	Windows Vista Home Premium
Notebook č. 3	Windows 7

Tabulka 5 - počítače použité k testování

5.1 Kompatibilita zařízení

Hotovou verzi zařízení jsem vyzkoušel na všech výše uvedených počítačích a u žádného se nevyskytl jediný problém. Zařízení je spolehlivě kompatibilní, jak s operačním systémem Windows XP, tak i s novějšími Windows Vista a Windows 7.

5.2 Uživatelská aplikace MFPlug

Testování uživatelské aplikace proběhlo na všech výše uvedených počítačích bez vážných komplikací. Pokud jsou splněna všechna kritéria, aplikace běží v pořádku na všech testovaných operačních systémech. Mezi tyto kritéria patří např. to, že v operačním systému musí být nainstalována podpora Microsoft .NET Frameworku 4.0[12], ve kterém je program napsán. Dále musí být nainstalován ovladač zařízení. Při splnění těchto podmínek jsem nezaznamenal během testování programu žádné problémy.

[12] URL: <http://msdn.microsoft.com/en-us/library/w0x726c2.aspx> [cit. 2011-07-27]

Více uživatelský přístup ke klientské aplikaci je řešen pomocí Isolated Storage (dále jen IS). Ukládání dat do IS jsem testoval vytvořením více účtů v jednom operačním sytému. Poté jsem se postupně přihlásil k těmto účtům. V každém jsem nastavil různé funkce, vzhled, jazyk a umístění programu MFPlug. Tedy informace, jež se ukládají pomocí IS. Při opětovném přihlášení zůstalo pro každého uživatele vše nastaveno správně.

5.3 Funkce uživatelské aplikace MFPlug

Všechny dostupné funkce programu MFPlug jsem jednotlivě testoval na všech výše uvedených počítačích. Všechny funkce fungovaly dle předpokladů a u žádné se nevyskytly problémy.

Funkce ovlivňující chod internetových prohlížečů byly testovány na všech počítačích a to na těchto prohlížečích: Microsoft Internet Explorer 9[13], Mozilla Firefox 5.0.1[14] a Google Chrome[15]. Na všech uvedených prohlížečích fungovalo vše správně.

5.4 Dálkové ovladače

Na většinu prací s programem MFPlug jsem používal jeden ovladač. Při testování jsem používal až devět dálkových ovladačů (Obrázek 38) najednou a uživatelská aplikace s tím neměla téměř žádný problém. Jeden se vyskytl, až když dva ovladače používaly stejné kódování, konkrétně to byl první a druhý ovladač z leva na obrázku. Program totiž ukládá kód infra červeného signálu, jež je vyslán dálkovým ovladačem při stisknutí tlačítka. Pokud by tedy byly dva dálkové ovladače kódovány stejně, měly by jejich tlačítka stejné kódy, proto by se aplikaci tyto ovladače jevily pouze jako jeden a ne jako dva různé ovladače. Kromě tohoto problému jsem však nenarazil na nic, co by bránilo správnému fungování programu.

5.5 Shrnutí testování

Testování všech částí mé práce proběhlo bez vážnějších komplikací. Zejména proto, že jsem většinu problémů odstranil již při samotné výrobě zařízení, implementaci uživatelské aplikace a výběru jednotlivých funkcí a jejich realizaci.

---

[13] URL: <http://windows.microsoft.com/cs-CZ/internet-explorer/products/ie/home> [cit. 2011-07-29]
[14] URL: <http://www.mozilla.com/en-US/firefox/new/> [cit. 2011-07-29]
[15] URL: <http://www.google.com/chrome?hl=cs> [cit. 2011-07-29]

6. Závěr

Během tvorby bakalářské práce jsem získal mnoho nových informací o různých technologiích a jejich využití v praxi. Hlavní cíl, tedy vytvořit přijatelné zařízení, a to jak po stránce velikosti výsledného produktu, tak i jeho ceny, které by bylo schopné přijmout infra červený kód z libovolného dálkového ovladače, jsem doufám splnil.

Neméně hlavní cíl, a to vytvořit k tomuto zařízení vhodnou uživatelskou aplikaci. Prvotní myšlenka byla pouze praktická, vytvořit program, který bude suplovat stisk kláves klávesnice, ovládání myši a také spravovat vybrané multimediální programy. Tuto myšlenku jsem později rozšířil a začal jsem také graficky zpracovávat vzhled této aplikace tak, aby byl co nejjednodušší pro uživatele a zároveň splňoval vše potřebné, tedy ovládání funkcí operačního systému z dálky pomocí běžných dálkových ovladačů.

Práce byla od počátku koncipována tak, aby se výsledná aplikace a zhotovené zařízení daly běžně používat v praxi. Navíc je možné volně používat také knihovnu MFPlugDll.dll, která slouží jako prostředník mezi knihovnou zařízení a libovolnou aplikací. To snad pomůže těm, kteří se budou touto problematikou v budoucnu zaobírat také.

Uživatelská aplikace se může také do budoucna rozrůst o nové funkce, modernizovaný vzhled apod. Avšak i v nynější podobě doufám program zaujme a bude využíván nejen mnou.

Seznam použitých pramenů a literatury

[1] MAYO, J.; *C# 3.0 Unleashed With the .NET Framework 3.5*. 2. vyd. USA: Pearson Education, Inc., 2008. 975 s. ISBN 978-0-672-32981-4

[2] TROELSEN, A.; *C# a .NET 2.0 Profesionálně.* 1. vyd. Brno: ZONER software, s.r.o., 2006. 1197 s. ISBN 80-86815-42-0

[3] BAYER, J.; *C# 2005 Velká kniha řešení.* 1. vyd. Brno: Computer Press, a.s., 2007. 813 s. ISBN 978-80-251-1620-3

[4] LIPNHARSKI, W.; *Infrared Remote Control.* [online], 2011, [cit. 2011-07-18]. Dostupné z WWW: < http://www.ustr.net/infrared/infrared1.shtml>

[5] ČESKO, I.; *IgorPlug: Ke stažení.* [online], 2003, [cit. 2011-07-18]. Dostupné z WWW: <http://www.cesko.host.sk/download.php>

[6] *Eagle online.* [online], 2005, [cit. 2011-07-19]. Dostupné z WWW: <http://www.elcad.cz/eagle/>

[7] MARUŠÁK, M.; *Výroba DPS nažehlením toneru.* [online], 2005, [cit. 2011-07-21]. Dostupné z WWW: <http://www.stepa.cz/katalog/slozky/vyroba_dps.pdf>

[8] *Izolované uložiště.* [online], 2011, [cit. 2011-07-22]. Dostupné z WWW: <http://msdn.microsoft.com/cs-cz/library/3ak841sy(v=VS.100).aspx>

[9] *The Winamp 5.02 SDK.* [online], 2004, [cit. 2011-07-25]. Dostupné z WWW: <http://forums.winamp.com/showthread.php?s=&threadid=168643>

[10] *.NET Framework 4.* [online], 2011, [cit. 2011-07-27]. Dostupné z WWW: <http://msdn.microsoft.com/en-us/library/w0x726c2.aspx>

[11] *Internet Explorer 9.* [online], 2011, [cit. 2011-07-29]. Dostupné z WWW: <http://windows.microsoft.com/cs-CZ/internet-explorer/products/ie/home>

[12] *Mozilla Firefox 5.0.* [online], 2011, [cit. 2011-07-29]. Dostupné z WWW: <http://www.mozilla.com/en-US/firefox/new/>

[13] *Google Chrome*. [online], 2011, [cit. 2011-07-29]. Dostupné z WWW: <http://www.google.com/chrome?hl=cs>